[LESP]

[Lessard Proposal]

On the Details of Computerized Videotape Editing

Theodor H. Nelson

17 July 1967

The Nelson Organization
Box 1546
Poughkeepsie, New York 12603

* * * * * * * * * * *

This note will discuss in detail certain aspects of a videotape editing console with a computer nerve center, designed to speed up videotape editing and enrich the editor's options. The general design was discussed in an earlier note, "A Sophisticated System for Videotape (and Motion Picture) Editing." In part this is an expansion of ideas presented before the Society of Motion Picture and Television Engineers in 1965, to be published in a larger version in the J.SMPTE later this year.

However, other contents are proprietary, and are here presented in confidence. Certain ideas in this paper may be patentable, and their enclosure in this proposal does not constitute their publication or relinquishment.

* * * * * * * * * * *

2

3

INTRODUCTION

It is now possible to put a computer at the nerve center of videotape editing, and therefore film editing as well. The details of switching and controlling tapes may be concentrated and consolidated in the computer, simplifying the editor's job. Individual shots will be automatically assembled into consecutive sequences, under the editor's direction. In addition, new services may be designed and programmed into the system, to simplify bookkeeping, render comparisons more convenient, and clarify decisions.

This is not the basic proposal, but an extension of it. The reader who wants a clear idea of the way the system is to behave should go to the section called "System Behavior and Services," page 30.

In this paper various details of such a system will be explained. These include 1) the control interface among the component units, 2) possible details of some operations, 3) the form of storage and filing, and 4) specific procedures that may be employed in shot rearrangement. In addition, there will be clarifications of possible services the system can perform, including such techniques as specifying shot assembly by written descriptions.

This paper describes a high-capability system, without shortcuts. Lesser systems are of course possible, and would be easier to implement. But in the long haul the system described here will be the important one, and it might as well be begun now as later.

There are three questions the reader is likely to have about the system described in this paper. First, can it be done at all? Second, is this method appropriate? And third, what is the exact usefulness of the services proposed here?

The answer to the first may surprise people, but is straightforward, an unqualified yes.

The second question, the appropriateness of the computer methods described-- especially the file methods-- is arguable among experts. It depends in part on the extent to which text handling should be a part of the system. It is my personal belief that without text handling the system can do nothing significant, except speed up what is done already.

It is on the thid question that the potential user may hesitate.

Of course, new and unaccustomed procedures are involved, and some of these may cause objections. One of the unusual features of this sytem is the use of text labels for commanding new shot sequences. Somebody has got to type these in. A few people I've talked to think typing is worse than splicing. But even if this is true, ways can be found of tying a secretary into the system. (Note that there is presently no way to go from dictation to computer text input.)

But users will not be always held to particular services that one person has dreamed up. These services are open-ended. They may be redesigned and changed at any time as better ones become apparent. This system can evolve and grow as we think of new services.

It should be emphasized that the user actions will be extremely quick and easy. Otherwise the system would be too much trouble.

A computer-based system shhuld always simplify and clarify human activity, making work easier and quicker. The popular image is unfortunately the opposite: that computers are extremely "difficult" and always "numerical." Neither of these is true. The system to be described here is intended to simplify and clarify. The user need never see a number unless he wants to use them.

Special pushbuttons, etc., will of course be needed. But these will not be discussed here. The reader must assume that when it says here, "the editor may do such-and-such," that a fairly convenient means can be provided for him to do it. But the exact layout of pushbuttons, etc., would evolve along with the system.

Other details to be omitted here include mass memory and other computer details. There will be some discussion of the switching hardware and procedures. But these, too, are shallow aspects of what has to be done. The problem is to provide comprehensive services involving text and alternatives. (If mere switching were the only problem, a ten-thousand-dollar computer could handle it.)

Much computer programming is necessary to make a basic system work. But as soon as a system exists which is better than present methods, improvements can be made at leisure. The user services are built on top and easily changeable, like the superstructure of a ship.

This paper is intended merely to state in detailed outline a set of technimques and services that will make such a system worth beginning, and persuade the reader that the system is currently possible.

CONNECTIONS

Although computer control of videotape recorders and switching is a new application, computer control of things in general is an old problem with old solutions. Control signals out from the computer, and feedback signals back from equipment to the computer, are a conventional matter.

The main input-output tasks need in this sytem are 1) exact control of videotape recorders (VTRs), 2) control of a switching network, and 3) text display. The mechanics of the text display will not be discussed here, since it is a standard computer application.

The digital interface is fairly straightforward, although there are certain choices to be made (exterior gating vs. holding registers, etc.)

Command code patterns would be sent out by the computer and be decoded so as to reach individual units as effective electrical signals. The decoding system is usually available from the computer manufacturer; specific connections to external circuitry must be established by the customer (or, in this case, the videotape engineer).

VTR Hookup

Computers can control videotape recorders (VTRs). The exact connections to be made to VTRs will depend on the technical details of the VTRs employed, and the relative simplicity and cost of particular connections.

Presumably the basic functions of the VTR are under electronic and/or solenoid control from the computer. These include Play, Record, and Rewind. They may also include Fast Forward, Mark (establishing an edit marker on an editing track of the tape), Step (move forward one frame), and Backstep (move backward one frame). Slow Forward, Slow Back and Transfer in Reverse (between VTRs running backward) would be particularly helpful for this application, but may be too difficult. It would also be beneficial to be able to change from Play to Record without stopping, and vice versa, and to go from normal speed to fast without stopping, and vice versa.

Presumably no VTR with all these faculties exists now. Whether it is worthwhile to have some of these faculties engineered, and which of them, will require detailed study. ~~The~~ Pencil-and-paper calculations of the time improvements each would offer should indicate, for each of these capacities, whether it is worth creating. (But as an example, I would guess that adding Transfer in Reverse would be as good as doubling the number of VTRs in the system.)

Switching Hookup

A large number of switching connections must be subject to separate control by the computer. Different signals by the computer must be able to connect specific video outputs with specific video inputs. In addition to the switching, the computer should be able to manage fades, dissolves and possibly mattes. For this reason controllable attenuators at the video inputs or outputs might also be wanted.

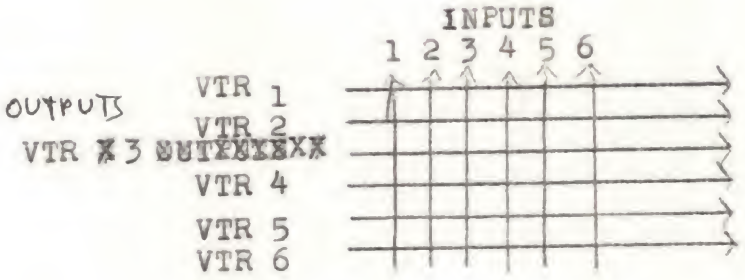
Similarly for audio. A complete audio switching system must be under computer control. Each VTR must be connectable by the computer to specific audio lines, and there must be a complete set of volume controls also under computer control. Continual reset of audio pots must be possible, for fully automatic control of the sound mix.

Equivalent switching procedures can be used for switching monitors and speakers.

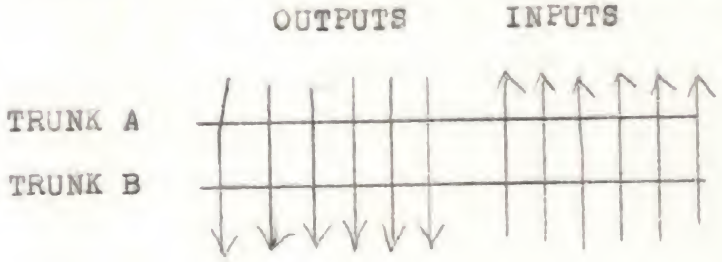
As it happens, video switching equipment which is digitally controllable is a catalog item from at least two firms (Whistler, Trompeter). The only questions are a) how to arrange the digital interface, and b) how to arrange the switches. The digital interface problem has already been mentioned. It is a design/cost problem, and does not require our attention here.

Matrix vs. Trunks

Another technical issue concerns the relative advantages of different switching methods. One method would be to have a complete switching matrix with crosspoints between all inputs and outputs.



A simpler arrangement would switch all inputs and outputs to and from trunk lines. (However, probably at least two trunks would be wanted, to permit setups in advance and simultaneous viewing of more than one monitor, if desired.)



The relative advantage of the trunk method increases with the number of VTRs, as can be seen from the number of video switching connections required by the two methods.

Number of VTRs	Matrix	A & B trunks (input & output from each machine to each trunk)
2	4	8
4	16	16
6	36	24
8	64	32
10	100	40

However, a counter-argument for the switching matrix is that a full switching matrix will permit unrestricted simultaneous copying among VTRs. The trunk method would require one trunk for each possible simultaneous transfer. If there were four trunks, no more than four transfers could take place simultaneously; and one of these trunks would be required for each monitor channel used, ~~th~~ removing them from participation in transfers.

VTR Status Signals, Timing and Feedback

Signals from the VTR to the computer will also be required. The selection of the most simple, practical or efficient signals will require detailed study.

One important class of signal would tell the computer the present activity of a VTR, so that its status could be checked. These could be pulses or voltage levels indicating the current motion (forward or back, fast or slow) and state (play, record, idle). A "frame" pulse, indicating the passage of a complete frame, is probably desirable; this could be gated by the flyback or blanking signals of the video. However, it should be independent of whether the VTR is going forward or backward, fast, slow or at normal speed. Perhaps there should also be a way for ~~this~~ the VTR to indicate that it has reached a destination-- beginning of tape, end of tape, or edit mark. (These might all be combined into a "completion" signal.)

A number of possible methods might be used for the timing of shots. It might seem obvious to use edit markers of the Editec type, as is done now. But this may not be as good an idea as it seems at first. The computer can by other means address to <u>any</u> frame, so that Editec marks are not essential. Moreover, when the exact beginning and end of a shot are not exactly decided, the Editec marks is no particular help; the computer can address the alternatives separately.

A variant scheme would distribute edit marks at regular intervals evenly throughout the videotape, and merely transmit special signals to the computer as they go by, instead of stopping.

This would mean that frame-by-frame counting would have to proceed for only short distances.

A third possibility is to use a cue track for complete addresses in digital format. The cue track used for Editec could probably be put to better use as an address track, holding the absolute address of every passing frame (as in Matsucka's system). This would save a lot of counting of frame pulses: the computer could always ascertain the absolute position of the moving videotape. It might also allow the computer to find out easily where a VTR has gotten to in a rewind or fast forward, without bothering to count every passing frame.

A full address track would be particularly advantageous if an address register could be installed on the VTR, so that the computer would not have to be continually interrupted to keep track of the progress of the different machines. This interrupt scheduling is the most difficult part of the whole system, and such an external search feature would be a great advantage.

How to use frame counts, edit marks and addressing depends on various statistical questions of count sizes and interrupt overhead. It also depends on stopping speed and stopping speed and accuracy of the recorder. If there is a danger of missing an occasional frame pulse because of the inertia of the recorder, such precautions as addressing become very important.

BASIC OPERATIONS

Large computer programs are made up of smaller ones.

Thus, to explain how to do a large job with a computer, it should be sufficient to explain how to do each small job of which the large one is composed. Then every time that part of the job needs to be done, the computer goes through that short program again.

It is the job of the computer programmer to lay out these tasks in advance, in great detail, so that they are waiting ~~xx~~ ready for the user, who needs to know nothing about how they work.

So far we have discussed electrical connections that will give the computer complete control of VTRs and their interconnections.

To the computer programmer, however, this assortment of hardware and attachments ~~are~~ is all alike: they are "devices" to be incorporated into the program through specific commands. By writing an instruction such as

#3 GO FORWARD

he can provide ahead for the occasions on which the computer must start VTR #3 going forward. By writing an instruction such as

#2 BACKSPACE Y FRAMES

he can provide ahead for the occasions that the computer will send VTR #2 backward a number of frames that will be

specified later.

Naturally, it is not these words that make things happen, but their coded equivalents, translated into electrical pulse-patterns. The words, though, can provide a convenient and exact way to list and plan the things that need to be done.

Here, in a similar form, is a short program. The computer should be able to send out coded equivalents of this program to accomplish an organized purpose.

```
CONNECT VTR 1 VIDEO OUTPUT TO LINE A
CONNECT VTR 4 VIDEO INPUT TO LINE A
CONNECT VTR 1 AUDIO OUTPUT TO LINE C
SET VTR 1 AUDIO OUTPUT TO HALF
CONNECT VTR 2 AUDIO OUTPUT TO LINE C
SET VTR 2 AUDIO OUTPUT TO 3/4
CONNECT VTR 4 AUDIO INPUT TO LINE C
START VTR 1 PLAYING
START VTR 2 PLAYING
START VTR 4 RECORDING
```

It will be seen that this set of commands makes connections that will, when the machines are started appropriately, perform a video transfer from 1 to 4 and an audio mix from 1 and 2 to 4. (That is, the starting settings for the mix.)

And, naturally, such commands as

```
CONNECT MONITOR M TO LINE A
CONNECT SPEAKER TO LINE C
```

may then be used to see and hear what is being recorded in the previous hookup.

Larger Combinations of Operations

The object now will be to demonstrate how small tasks, like those described above, can be combined into complete and useful actions. We will look at some significant jobs, essential to an editing setup, and the way they are built up of smaller sections. (The reader must accept on faith that each instruction listed here can in turn be built up from the basic operations the computer will have available.) The procedures described are not necessarily the most efficient; they are intended to explain.

1. HOW TO COPY FROM ONE VTR TO ANOTHER.

Connect VTRs appropriately to same line
Give PLAY and START commands to source VTR
Give RECORD and START commands to target VTR
Begin counting frames; each time a frame pulse arrives,
interrupt current program and subtract one
from the frames left to go
When the number of frames left to go reaches zero, stop.
(Alternatively: set timing register and compare it
repeatedly with a 60-cycle clock, till it matches
or exceeds time shown at clock.)

The following methods use lists of shots and sequence, stored on a disk file. (Storage methods are described later in this paper.)

2. SORTING METHOD.

(This program of actions transfers a series of shots from VTR 1 to VTR 2, putting them in a new desired sequence.)

Create list of shots to be transferred ("assembly list") in order

Ascertain length of whole sequence by adding up lengths of component shots on assembly list

Move VTR 2 to beginning of blank tape

Make sure length of whole sequence is not greater than available blank tape on VTR 2

Go to very beginning of assembly list

Ascertain "next" shot on assembly list

Fast forward or backspace on VTR 1 to beginning of that shot

Transfer that shot, recording from 1 to 2

Test whether all shots on assembly list have been transferred

If not repeat program loop

If so, stop

3. PROCEDURE FOR FINDING A SHOT ON VIDEOTAPE.

(This assumes that the shot is stored somewhere on VTR #5.)

Look down the list of shots now stored on VTR 5, testing whether each is the desired shot number

When and if that shot number is found, ascertain its address on the tape

Subtract the current position from that address

Go forward or back the resulting number of frames, the direction depending on whether the result is positive or negative.

Example. Suppose the shot you want begins at frame 900. The VTR is currently positioned at frame 1450.

Desired position	900
Minus current position	-1450
	- 550

The VTR is then backspaced 550 frames.

As has been stated, these have been examples for explication, and leave out important programming techniques. Moreover, the methods used for sorting and allocating tasks and space among VTRs will need considerable analysis and improvement. This falls within the realm of "optimizing," and is not relevant to this paper.

HOW MANY RECORDERS?

An important question is how many VTRs to use in such a setup. The more, the merrier; but they cost a lot of money. The system could really not do very much with fewer than three; but one feels instinctively that over five would be extravagant.

However, having many VTRs will make possible "overleaved" transfer: while the editor is looking at one thing, the computer can be setting up something else entirely on machines not currently involved. As a guess, I would say that six VTRs would permit the editor to view almost continuously when he is not picking or typing.

It might be that three or more top-quality VTRs would be used for the masters and the final copies, but economyw models with less fidelity could hold parts of intermediate assemblies.

The problem of how many VTRs is easily settled from the point of view of computer software: the computer program must be able to function similarly regardless of how many VTRs are available during a given session.

INFORMATION STORAGE

The preceding notes have shown the way that particular operations of the system may be carried out, step by step.

A more fundamental problem, however, is organizing the overall task to be performed, and handling the information-- numbers, text and relations-- that locate and describe the shots stored within the system.

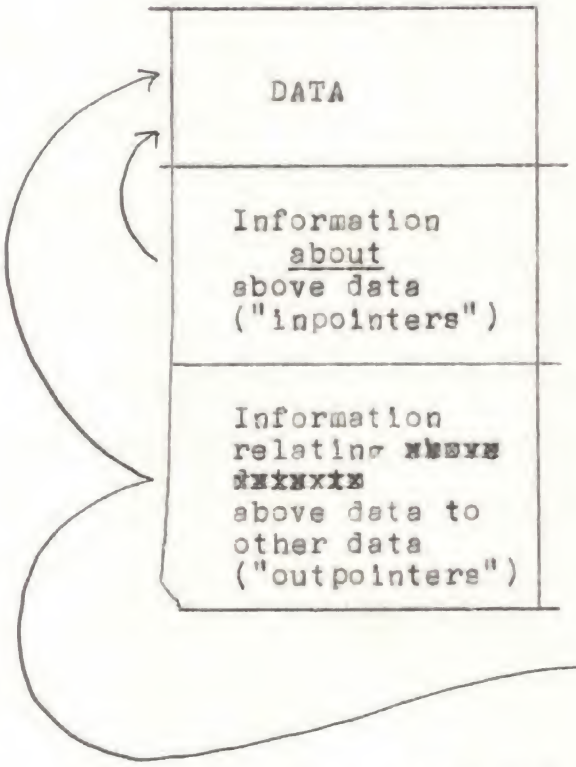
These things are complicated. Not only that, but they represent a change in the way computers are to be handled: an opening up of the machine to more irregular kinds of information, rather than reducing the information to the structures that machines ordinarily handle.

Since 1960 I have been working on the problem of how to store information of this kind, the annotating and cataloguing of irregular materials with texts of irregular length. This work has been a development of the list processing and "pointer" techniques developed by computermen in the last few years. These techniques allow stored data to refer to other data in a grossly irregular fashion.

The result of my work is a type of data structure which I now call POIGNANT, capable of handling all this information and more. (The structure is presently being implemented for the IBM 360 at Brown University, under my direction.) Specific programs need to be created for unpacking and using the data, but that is much less of a problem now that the basic structure is well defined.

POIGNANT data may come in units of any length, and be interconnected any number of times, in any number of ways.

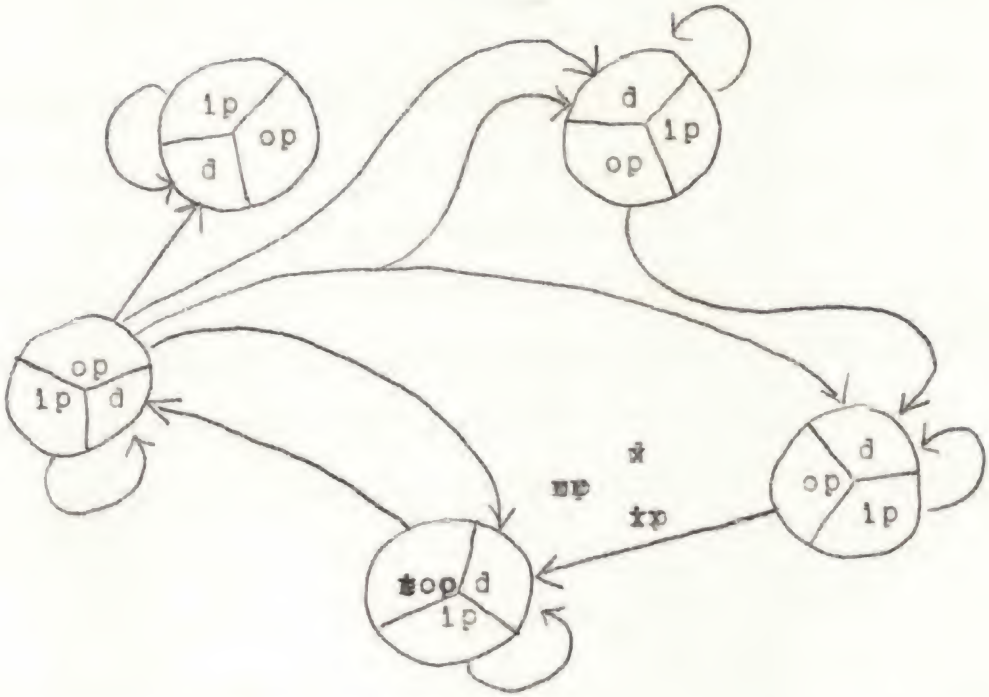
The POIGNANT structure holds data in "trains" of standard-sized blocks, with extra information <u>about</u> the data stored separately ("inpointers") and connectors to further data elsewhere in still a third package ("outpointers").



POIGNANT "TRAIN"

(Many technicalities omitted)

Various kinds of data and meta-data are defined in these terms and interconnected to any degree, as suggested by the following illustration, suggesting relational connections among data packages.

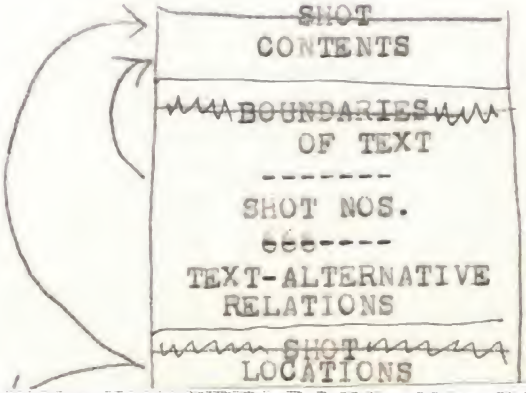


We may consider this as a new method of storing data having any form whatsoever.

In the videotape system described in this proposal, a large problem is the changing data, which can assume very complex forms and structures. The changing lists of the contents of each VTR, and most difficult, the storage of names and descriptions of shots, can be stored and continually revised in this datax structure and its servicing programs.

While the details of this storage are not crucial here, a few illustrations will indicate approximately the way these compound structures will be held.

The list of shot contents:

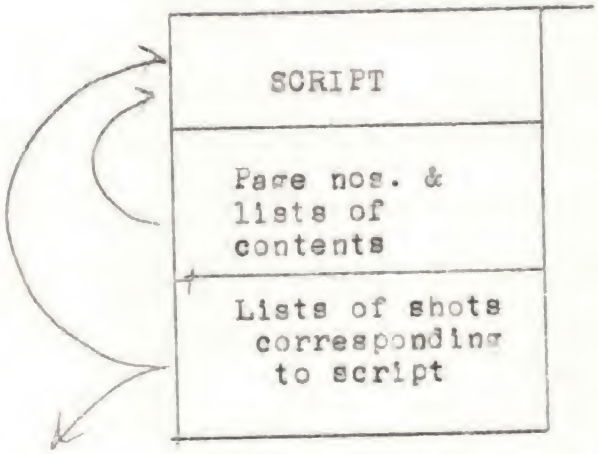


(Data)

(Inpointers)

(Outpointers)

The script and its interconnections:

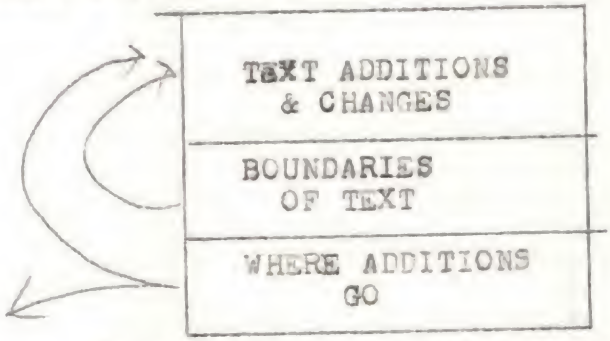


(Data)

(Inpointers)

(Outpointers)

Revisions of the above materials that have not yet been spliced into the main lists:

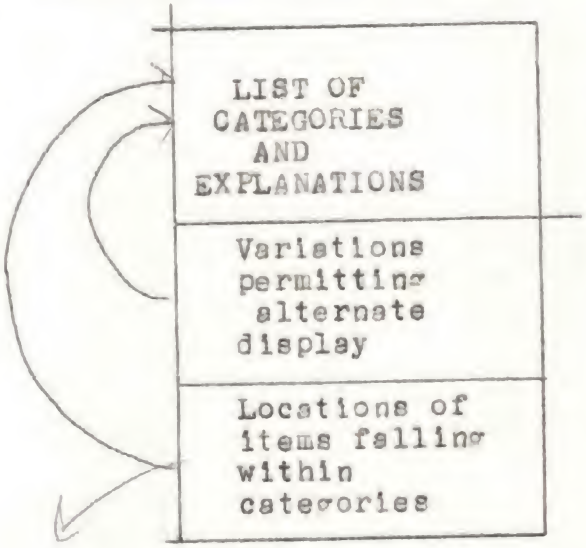


(Data)

(Inpointers)

(Outpointers)

Categories among which material is to be cross-indexed:



(Data)

(Inpointers)

(Outpointers)

LISTS

Many different kinds of lists of information must be stored in the system, to keep track of shots on tape and other information. POIGNANT makes it possible for these lists to be irregular and complicated.

High-speed core memory of the computer is quite limited in size. For this reason it is necessary to store long lists on larger memories which arex outside the computer proper. In particular, disk files are most often used for this purpose.

In this system, then, lists of shots and text would be principally be kept on a large disk file, or on magnetic tape. Some lists would be stored in an ordinary fashion, some in POIGNANT. These lists will have to be interconnected and s stored in ways I won't begin to describe. (These may not necessarily all be separate, but might be combined in certain ways for convenience.)

1. ORIGINAL SHOT LIKST

This list gives the original shot numbers and the addresses of their beginnings on the source videotapes.

REEL NUMBER	Current VTR No.
1	BEGINNING ADDRESS
2	BEGINNING ADDRESS
	·xx · ·
SHOT NO.	BEGINNING ADDRESS
SHOT NO.	BEGINNING ADDRESS

2. CUTTING-POINT LIST

This one tells the cutting-points, or cutting-point ranges, which the editor is considering for each shot. There are several situations, of which at least ~~four~~ three should be recognized within this sytem.

a. exactly defined shot

b. alternative cutting-points

c. cutting-point ranges

We will call these three possibilities cut types, and mark them accordingly. Different editing procedures will treat them differently. ~~accordingly~~ (see "User Services").

CUTTING-POINT NO.	CUT TYPE	START OR FINISH?	LOCATION OR RANGE

3. LIST OF DERIVATIVE SHOTS.

These are derived, of course, from the different cutting-points.

ORIGINAL SHOT NO.	NEW SHOT NO.	CUTTING POINTS: START	FINISH

4. ~~XX~~ CURRENT MASTER SHOT LIST.

This one tells the location of every shot the editor is interested in, on each VTR.

SHOT NO.	GEN-ERATION	VTR NO.	ADDRESSES: FIRST FRAME	LAST FRAME	(Original) (Copy 1) (Copy 2)
	1				(Original)
	2				(Copy 1)
	2				(Copy 2)
			. . .		

5. CURRENT DIRECTORY OF EACH VTR

VTR NO.	
SHOT NO.	STARTING FRAME
~~XXX~~ 273 ~~X~~ . . .	1 . . .

~~SHOT NUMBER/SHOT TITLE/DESCRIPTION/COMMENT LIST~~

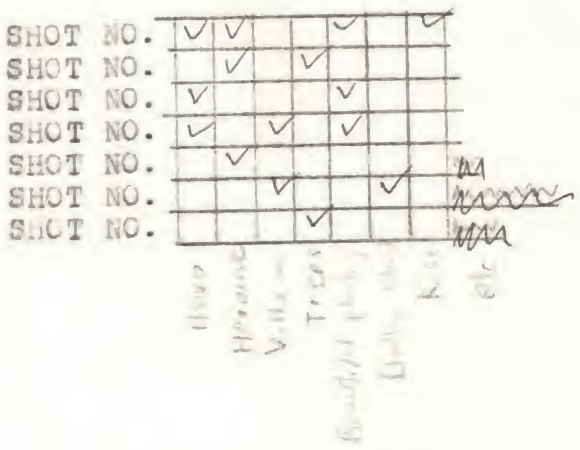
6. SHOT CONTENTS LIST

This contains shot titles, descriptions and comments, including alternative and abbreviated descriptions or captions.

SHOT NO.
SHOT NO.
SHOT NO.
SHOT NO.
SHOT NO.
SHOT NO.
SHOT NO.

(Changes in the contents lists may be made at any time.)

7. SHOT CATEGORY LIST (simplified)



Cross-indexing under all these categories is possible. (See "System Behavior and Services.")

8. SCRIPT LIST

The script list, if any, states where in the script shots go and to the extent that, (if the editor cares to bother). Types of inherent uncertainty may be observed. If a shot might go in several different places, or contains events in other than their proper script order, "script pointer codes" may be developed which correctly represent this. (This will permit the editor to request, for instance, all the shots that "could" go in a certain place--according to his own previous judgment.)

The following illustration shows the ways in which shot information can be made to correspond to the script in a list of this type. (Detail can be down to the individual word in a script, but presumably that will rarely be needed.)

SHE:

You're a creep.

HE:

You're another.

SHE:

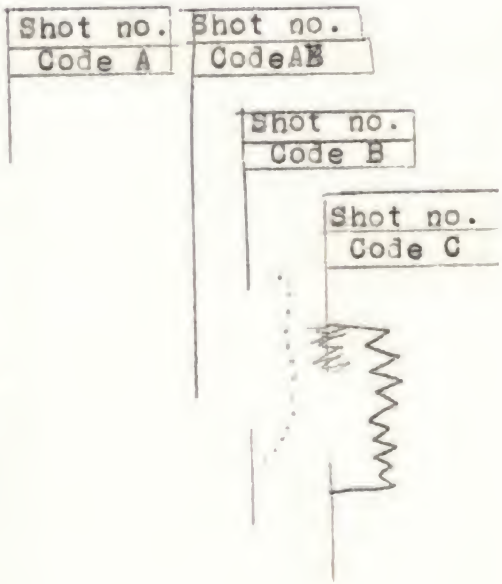
Ah, your father's mustache.

HE:

Same to you.

SHE:

Think you're so clever.



(In the case of the third shot, the director had the actor speak both lines in the same shot, but with no pause, so a cut would have to be made. That is what "Code B" means. The fourth shot, which shows her reaction, could be used in either location but not both. That is what "Code C" means.)

9. OVERLAP LIST.

(This list would indicate what shots overlap according to the script list.)

10. ASSEMBLY LISTS (LISTS OF SHOTS TO BE ASSEMBLED)

11. LISTS OF SHOT SEQUENCES FINISHED AND NAMED.

These basic lists would constitute the basic data of the system. They would be added to and changed as the editor added and changed information and the contents of different VTRs changed.

These are the basic lists required for shot-by-shot assembly. (It will presently be explained how sequences of shots may be assembled.)

Any number of additional facilities can be constructed using these lists. What they should be, however, is extremely hard to predict before the basic system is constructed. Users will have to say what they want after experience with the basic system. As more and better user functions are requested, however, more lists may need to be added. ~~However, examples~~

However, examples can be given of modes of work that should be useful, and begin to show the power of such a system. These modes of work are well-defined and can be built into a basic system immediately.

SYSTEM BEHAVIOR AND SERVICES

The purpose of this system is to speed up, simplify, clarify and enhance all the operations of videotape editing, without introducing any restrictions or distractions. All the detailed manipulations are to be performed by the computer according to an evolving list, inside it, of the shots and sequences the editor is interested in.

The general sequence of operations is as follows. The editor views the original videotape and types names and comments into a keyboard (or dictates them to a secretary, who types them in). By punching buttons at the start and stop of the shots that interest him, the editor informs the computer where the shots are that are to have these names and comments attached.

The editor may now call to the screen these names, and point at the ones he wants combined into sequences. As he points at these names with a special device, they appear in a new list on the screen. He may move these names around till the sequence seems right.

The sequences are now prepared rapidly and automatically according to the lists he has ~~proposed,~~ prepared, and they are soon ready for viewing. He may continually revise the lists (and make variations of them), and have the computer show him the resulting sequences quickly. And he may work over the details of splices and cutting-points carefully until he is done.

That is the basic system: view, select, trim and reconsider. Repeat in any order until done.

If necessary, the editor may ask for a second-generation final copy. The computer will then arrange to copy every shot from the original tape.

In what follows we will discuss details of some of these services. However, it should be remembered that many other services are possible within a system of this kind, and these services are only a beginning.

The appearance of the words on the screen, and the way they will move, depends on the form of text display chosen. The better systems offer lower case letters as well as capitals, and will permit phrases to move slowly, as well as simply jump. This capacity is preferable, so the editor can always see exactly what he is doing.

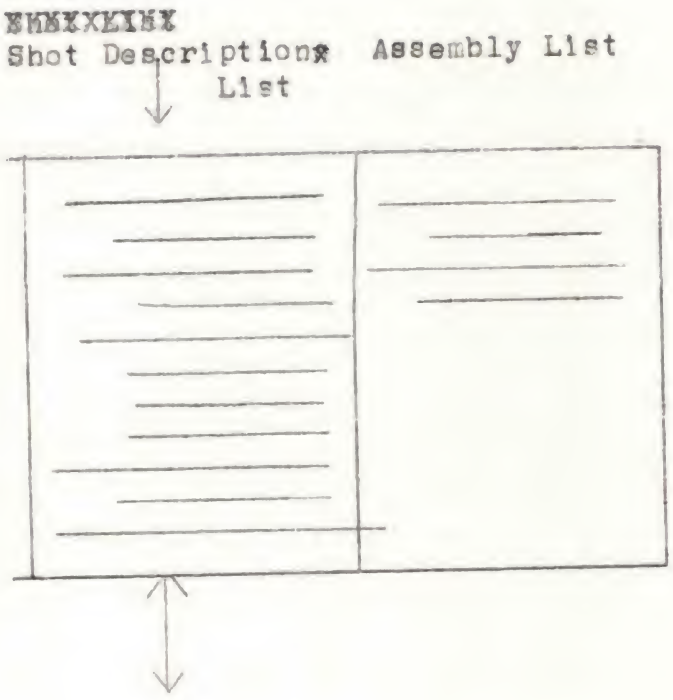
Controlling Shot Rearrangement

This section will describe in more detail the service of shot rearrangement by title or description.

The interior means by which the sorting of phrases causes the sorting of video will not be discussed in detail here. Essentially, when a shot description is chosen, the computer examines its number, looks it up its number in the shot address table, spaces appropriate VTRs forward and back, and transfers the shot accordingly. The details are black magic (programming).

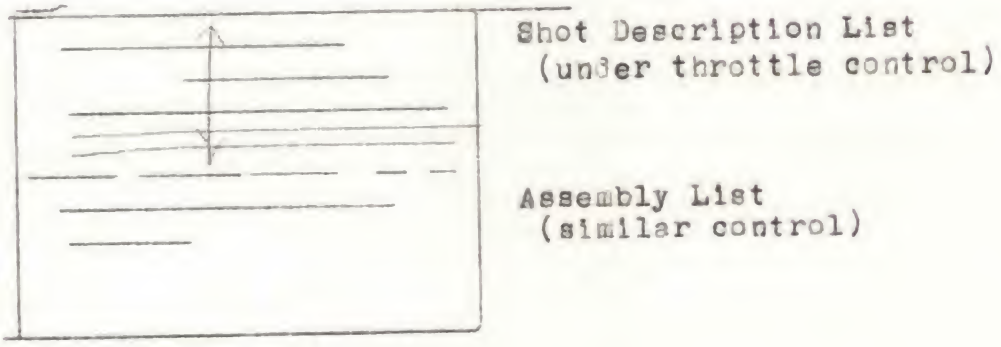
"Pointing" to aspect words on the screen would probably not be with the usual light pen, which ordinarily does not work with a TV screen, but rather an Engelbart Mouse, which is a small box on wheels which you roll around the desktop. This instrument is one of the simplest and cheapest pointing devices, and is easy to install.

One possible arrangement would divide the screen into two vertical sections. The left section could look into a list of the shots. Although only part of the list could be be seen at any given moment, the user would have a throttle permitting him to "scroll" up or down the list. The list would then move backward or forward on the screen like a drum titler, with new items appearing at the bottom edge as the old ones disappeared at the top-- or vice versa.



33

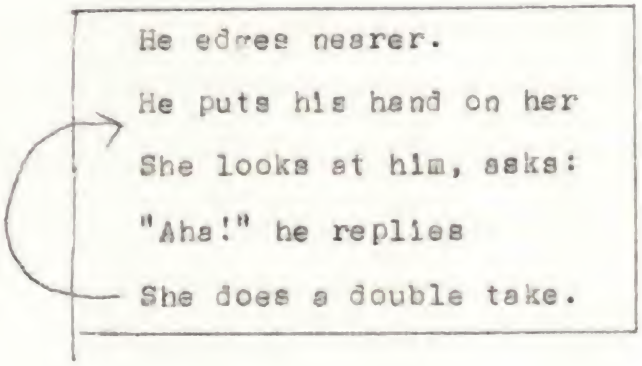
The obvious alternative is to divide the screen vertically:



The other half of the screen would hold the evolving assembly list. The editor would point at the item he wants in the shot description list, it would blink acknowledgment (so he could make sure he had pointed correctly), and then he would point at the place he wanted that shot inserted on the assembly list. In a flash that item would be added to the

the assembly list; and, if it had been placed between other items, those other items would have moved apart to make room.

He can then examine the assembly list. If something is in the wrong place, he can move it around with the pointing device, until it is ready.



Editor points at bottom line.
It blinks.
Editor points between second and third lines.
Text can either jump, or move through other text.

34

The choice of shot titles can be quite free, and very versatile services can be designed around them. (Of course, many x issues of taste are involved. Some editors may want to work with numbers. This is permitted too.)

Let us consider some flexible aspects of this title-manipulation system.

The titles of shots can have details that are useful or distinguish between them:

SHE STEPS TOWARD HIM (HAIR STREAMING)

They can contain information on photography:

THEY FACE EACH OTHER (PAN SHOT)

THEY FACE EACH OTHER (TWO SHOT CU)

However, you may not necessarily want all that information on the screen when compiling assembly lists. The editor may want to manipulate short titles on the screen but have more written details available. Suppose a shot had the following full title:

THEY WALK. HE LOOKS HAPPY; SHE LAGS A LITTLE, APPREHENSIVELY

When desired, the editor could have this reduced; for example, to:

THEY WALK. *

or

THEY WALK. * SHE LAGS *

(The asterisk reminds the editor that he has more details in the title if he wants to see them.) At different stages of the editing process, different phrases of the shot descriptions can be included in the screen caption.

An alternative method is simply to cut off the shot description at the end of the first line. Obviously the editor may have his choice of how to shorten the shot descriptions.

Shot descriptions can overlap in various ways. For example, all the different shots of the same event may have a common description, with qualifying phrases or codes alongside. If an actual script is included in the system, the tie-in of that script will also involve considerable overlap among shots.

Not only individual shots, but assemblies of shots, can be given names, and be transferred to tapes automatically by the user's placing these names on an assembly list. Such conglomerate names might for convenience end in "scene," "sequence" and "number," e.g., "Love Scene," "Chase Sequence," "Chorus Number.")

It will therefore be possible to create the whole work from finished scenes by simply listing the scene names and giving other instructions that command a final copy to be made. For instance, a commercial can be ordered into existence simply by naming its parts (which have already been finished separately):

BEACH TEASER
BOTTLE POURING
SPLASH MONTAGE
DANCING
BOTTLE/GIRL MONTAGE
SUNSET, WITH PRODUCT

Just because a shot or sequence has been described and listed does not mean the editor has to use it. The editor may list for assembly all the shots in a given category, or all the out-takes in a given scene. He may also create different assembly lists for the same scene, giving them slightly different titles, in order to examine possible variations.

~~It will also be possible for the editor to request out-takes,~~

~~in a given scene. He may also create different assembly lists for the same scene, giving them slightly different titles, in order to examine possible variations.~~

It will also be possible for the editor to request out-takes, leftovers and other collections of shots without assembly lists, as long as an exact definition of the shots he wants can be framed in terms of their categories or assembly status.

The editor may also request shots that have a certain connection to the script list, or particular connection codes.

~~One shot can be divided up different ways, and the resulting clips can be given different names. (But the editor may also request shots that have a certain connection to the script list, or particular connection codes.~~

One shot can be divided up different ways, and the resulting clips can be given different names. (But the editor will always be able to check on whether he is using the shot elsewhere.)

It will be possible for the editor to inquire of the system whether he has repeated any shots, or parts of shots, in a given assembly. Other related types of question will be possible: for instance, "Have I changed the original sequence of any of these shots?"

When there is a fixed script in the system, it can be used to provide a quick written summary of the cutting. The editor would ask the computer to display the shot descriptions in a specific assembly next to a script:

SCRIPT	SHOT DESCRIPTIONS IN PARTICULAR ASSEMBLY
(We see the latch slowly raised. The door opens. A nose appears.)	CU latch Perespective ~~xxxx~~ ~~xfxxxxxx~~ shot of nose. Pan to the waiting gunman.
RANDY Gee, a nose!	Two shot of RANDY & UNCLE BILL.
UNCLE BILL Don't be too sure, son.	

Of course, the script correspondence would have to be logged into the system, in a manner similar to the handling of categories; (unless it could be done in advance according to some sort of production manager's record). But once the original shots had all been logged against the script, in sufficient detail, the system could automatically match the shot descriptions for an assembly to the script.

The use of text on the screen is not the only possible method for keeping track of shot information, although it is perhaps the most versatile.

Another possible scheme would put selected frames from specific shots into some device on which they could later be reviewed quickly-- either a VTR or a video disk. The editor could see them immediately to refresh his memory without seeing whole shots. (They could be overlaid with symbols and arrows

to indicate motions, and with comments, etc.)

It would also be possible for the editor to dictate his comments directly into the system. They would be stored on some addressable tape device, possibley a VTR, and be heard along with, or before, the shots during which the dictation occurred.

Unfortunately, this would not lend itself to painless re-sequencing the way xxx that the written text in the computer does. However, it might be a good method for handling a lot of minor comments worth reviewing.

Line drawings can be included, and allow the editor to store, and superimpose, rough animations of the compositions he wants to try, in sequence. This sketching facility would come "free" with certain machines, such as the DEC 338, but considerably increase the trouble of programming the system on others, such as the LINC-8.)

CATEGORIZING

When he types in the shot descriptions, the editor may also choose to put the shots in categories. The categories are to be of his own ~~xx~~ selection, of course. Shots can be indexed on many categories, ~~x~~ including quality. For instance:

Subject

Hero
Heroine
Villain
etc.

Quality of composition

Magnificent
Good
Fair
Acceptable
Lousy

Movement type

Still
Camera movement
Subject movement
Background movement

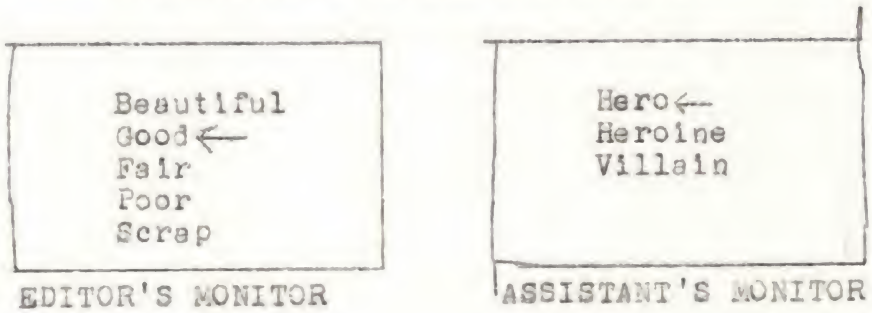
Movement speed

Slow
Medium
Fast

40

Assigning shots to these categories requires no tricky abbreviations or coding forms. After each shot the list can roll by on the screen, and the editor can simply point at the attributes it has. For simpler attributes, such as who is in the shot, these matters could be decided by an ~~xxxxxxxxxx assistant.~~ assistant at another monitor.

SIMULTANEOUS CATEGORY DISPLAYS



To save time in some applications, the category lists could be superimposed on the passing picture.

The editor may ask, then, to see all shots having any of his categorized properties, or any combination of these properties. (It is wise, however, to begin by asking the system how many there are having this combination. Bad guesses can waste time.)

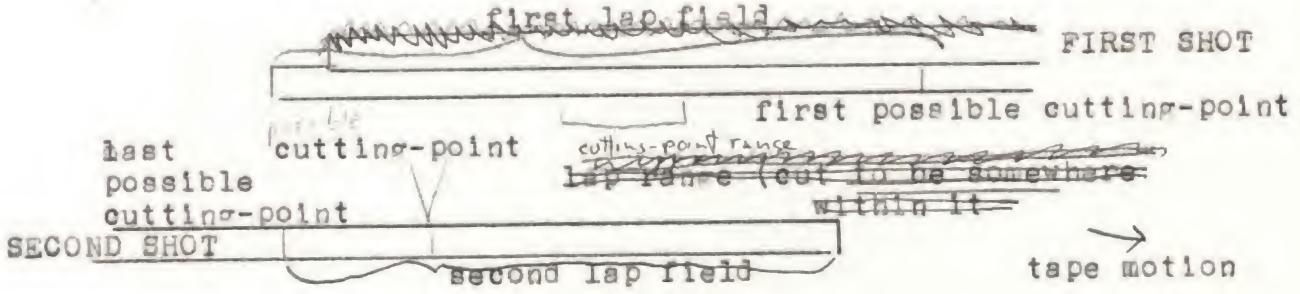
LAP CONTROL

One of the problems in film editing, and presumably video editing as well, is deciding where to make a cut between consecutive shots. Actually this is two decisions: where to end the first shot, and where to begin the second. (Exception: where they are synchronized to a common sound track or timed to some common event-- let us call them "synched" shots-- then there is only one decision to make. But where the exact timing is optional, then there are two decisions.)

Let us call this the problem of "lap control." The system under discussion can offer services to ease the editor's task when decision is difficult.

The following diagram suggests terms which will help discuss the issue.

"LAP PAIR"



In the system under discussion, the editor specifies to the system through pushbuttons that two particular shots are consecutive. He then views the shots, and specifies the cutting-points he would like to try, or the ranges within which he thinks

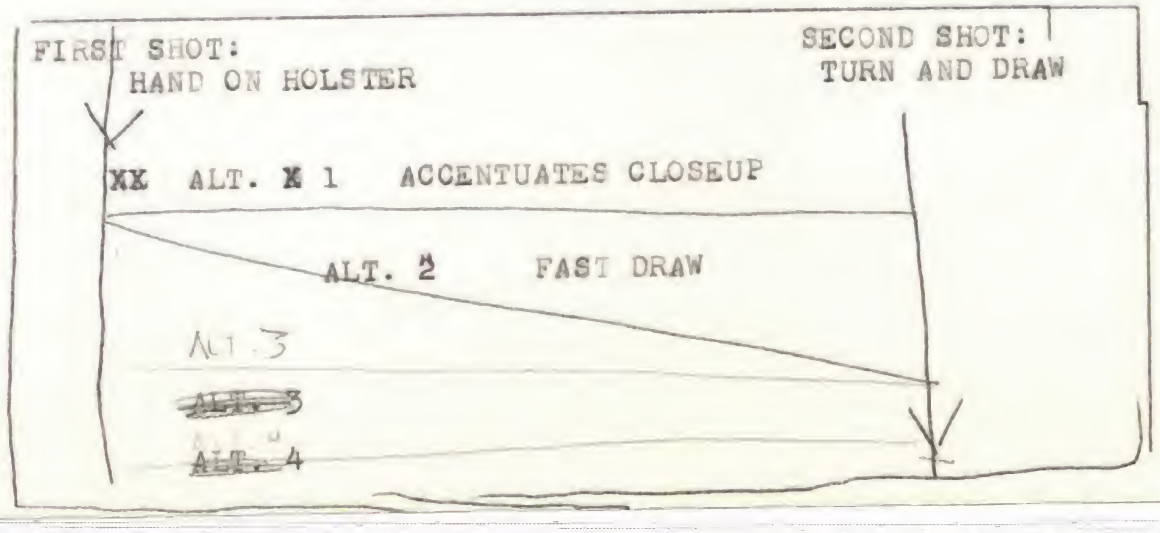
he would like to cut. He may, in addition, type in or dictate remarks on the different cuts.

After a pause, the system is ready to show him the several lap alternatives in quick succession. Before each lap pair, his comments on that alternative are displayed on the screen or played through a speaker. The closing part of the first shot is shown, and then the first cut he has requested is seen. The system is immediately ready to show the next lap pair, then the next, and so on. (It should be pointed out that the editor cannot usually ask to see "all combinations" of the cutting points in the leading and trailing shots, since the number of shots may grow astronomically. If the editor is considering five cutting-points on the leading shots and five on the trailing, there are twenty-five possibilities.)

To make slight adjustments in the cutting-points, special adjustment thumbwheels or other custom controls can be provided. The lap pair would then be seen again.

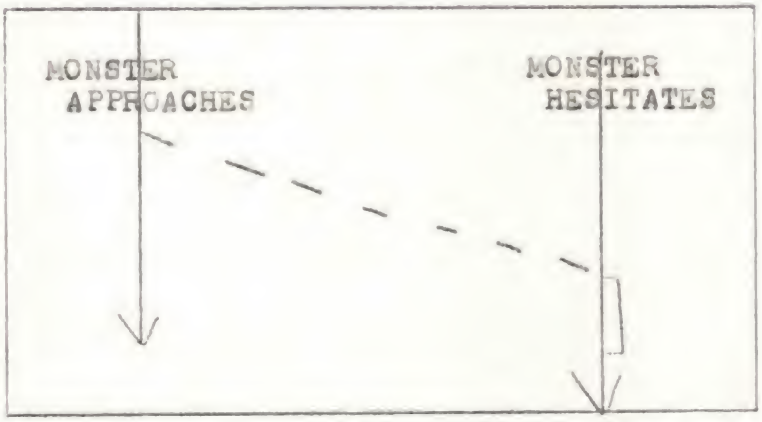
Another possibility would be a video display such as the following:

"LAP MAP"



This is a map showing six different cutting-points, and four alternative possible "splices" between them. The editor has written comments on two of them. He may look at the four alternatives in quick succession, then modify any or all of the alternatives by calling this map back to the screen and performing operations upon it with the special controls. (For instance, he may slide a cutting-point on the xx map and the cut will be changed accordingly.)

A similar technique would help select a cutting-point within a lap range. Suppose the editor has chosen a lap range:



The two shots can be shown <u>superimposed</u> for the duration of the lap range, or a dissolve can take place, beginning at the start of the lap range and ending at the end of the lap range. The editor may make a quick adjustment on the lap map, and immediately see the resulting cut.

I should point out that to make such an adjustment on the lap map should take no more than ten seconds.

The reader may doubt the immediacy of successive replays in the above description. For this reason, let us examine in detail a possible way of handling the lap-control video.

Let us assume that the original is on VTR #1, and there are four other VTRs, 2 through 5. The editor specifies two shots which are to be consecutive, Shot A and Shot B.

VTR 1 is moved to the beginning of Shot A.

VTRs 2 to 5 are moved to free stretches of tape, or areas which may be overwritten.

Shot A (in its longest version) is copied simultaneously from VTR 1 to all the others.

VTR 1 is now moved to the start of Shot B.

Shot B (in its longest version) is copied simultaneously from VTR 1 to all the others.

VTR 2 and 3 backspace to the beginning of Shot A.

Shots A and B are now copied consecutively from 2 and 3 to 4 and 5.

VTR 4 and 5 now backspace over all four shots they have copied.

All four shots are now transferred from 4 and 5 to 2 and 3. The lineup is now:

VTR 2	A	B	A	B	A	B
VTR 3	A	B	A	B	A	B
VTR 4	A	B	A	B		
VTR 5	A	B	A	B		

We now have more than enough shot pairs for a new pair always to be ready. (Other arrangements may be preferable.)

It is now a relatively simple matter to show lap pairs

consecutively with different timings, backing up VTRs individually as needed. The computer switches between VTRs at specific cutting-points as required.

FUTURE SERVICES

The services that have been described are intended to be a beginning, good enough to start on, and in fact worth starting, which is important.

But this is just the beginning. Many more services are possible; some might be excellent.

Yet the design of these services is a difficult matter. On the one hand, computers cannot do everything, and it is very easy to ask too much. On the other hand, judicious and artistic choice is the human prerogative, and it is easy and sometimes tempting to give it up to the computer, which is certainly a mistake.

In any case, let us consider a few more services which may lie somewhere between the impossible and the undesirable. These constitute speculation and are not part of this proposal.

One way this system can be extended is to handle more complex relational notations than have been described. Complex services could be built around these relations. Such relations might include "This shot could be used in several different places, but only if..." and "This part of the shot is unusable, but the parts around it are good."

Another second-generation service would be what we may call Partial Specification. This is a fairly complex application, which needs explaining.

It would be handy to the computer programmer if an editor could always specify the exact sequence of shots that he wanted. However, subtle methods may be possible that exactly represent

the vague choices and relations that x really exist in his mind. For instance, the editor might specify relations that existed between two shots, such as "this comes somewhere before this" and "if I use this I can't use that." We may call this procedure, building up vaguely from relations between pairs and small bunches of shots, partial specification.

By specifying such relationships one at a time, the editor can gradually build a file of guiding information in computer storage, working his way toward the eventual exact sequence. The system could tell him at any time whether the set of relationships he has specified yet determine an exact sequence, or whether any of them are contradictory. If they are contradictory he may take some of them back, or sort them into different sets of relationship premises.

Other services would permit the user deep organization of his thoughts in visible form. It should eventually be possible to allow the user to consider complex alternatives, log them and intercompare them in detail. A system of this kind was discussed in my paper "A File Structure for the Complex, the Changing and the Indeterminate."

If no system like these exists today, it is because subtle creative applications of this type have so far been avoided by systems analysts, and not because there is anything inherently impractical about it. However, this aspect of such a system would be comparatively experimental. It will take a lot of work to devise functions like these that please the user but are logically analyzable by the machine. Such aids are for the next generation of systems, the son of this one.